## **Load Balacing & Session Clustering**

를 날짜	@2020년 12월 21일 → 2020년 12월 22일
≔ 태그	

### → nginx + tomcat (docker) load balancing

- 1. nginx load balancing setting
  - → upstream : 각 포트별 로드밸런싱에 사용될 포트를 지정

```
http {
    upstream http_stream {
        server 127.0.0.1:8080;
        server 127.0.0.1:8081;
    }

    upstream https_stream {
        server 127.0.0.1:8443;
        server 127.0.0.1:8444;
    }
```

→ HTTP setting

```
server {
   listen
                 80;
                 [::]:80;
   listen
   server_name
                 /usr/share/nginx/html;
    root
   # Load configuration files for the default server block.
   #include /etc/nginx/default.d/*.conf;
   location / {
            #proxy_pass http://localhost:8080;
            proxy_pass http://http_stream;
            proxy_set_header Host $http_host;
   }
   error page 404 /404.html;
        location = /40x.html {
   error_page 500 502 503 504 /50x.html;
        location = /50x.html {
```

### → HTTPS setting

### 2. 각 tomcat별 로그 확인

testOne, testTwo URL을 80포트에 접속하게 한 뒤 로그가 각각 어디에 찍히는지! nginx 80번 포트를 통해 URL을 입력했지만 docker로 올린 각 tomcat에 로그가 찍히는 것을 확인할 수 있다.

tomcat #1

172.17.0.1 - - [21/Dec/2020:09:07:51 +0000] "GET /login.do\_test0ne HTTP/1.0" 404 732

tomcat #2

172.17.0.1 - - [21/Dec/2020:09:07:55 +0000] "GET /login.do\_testTwo HTTP/1.0" 404 732

### → Session Clustering

https://jistol.github.io/java/2017/09/15/tomcat-clustering/ 위 링크를 보고 아주 간단하게 세션 클러스터링을 설정할 줄 알았지만 역시나 실패

 $\rightarrow$ 

8443, 8444 포트로 접속했을 경우 세션 ID 동일한 것 확인

### → <distributable/>

해당 구문을 web.xml에 추가하게 되면 setAttribute 부분에서 에러가 발생한다

```
java.lang.IllegalArgumentException: setAttribute: 직렬화할 수 없는 숙성 [loginStatus] at org.apache.catalina.session.StandardSession.setAttribute(StandardSession.java:1416) at org.apache.catalina.session.StandardSession.setAttribute(StandardSession.java:1376)
```

### http://blog.naver.com/estern/110011499088

톰캣은 재구동 시에 세션을 유지하도록 설정되었고, 해당 설정은 default로 적용된다. 이 설정은 톰캣 종료 시 세션 정보를 직렬화하여 시스템 파일에 저장하고 다시 로드하는 방식으로 작동한다. 따라서 세션에 담기는 모든 attribute는 직렬화하여 저장된다.

## https://tomcat.apache.org/tomcat-8.0-doc/cluster-howto.html

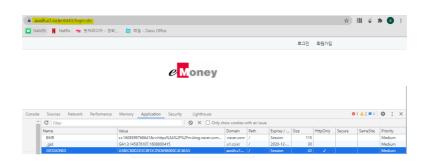
위의 링크에 따라 진행한 두가지 사항 외에 반드시 필요한 것이 하나 더 있었다.

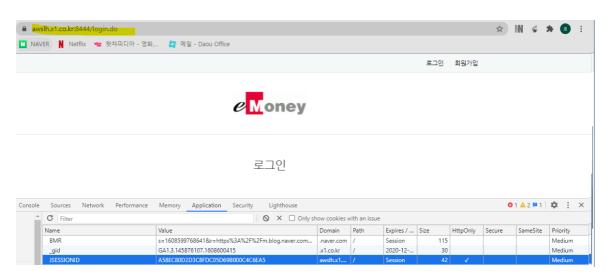
- 1. Uncomment the Cluster element in server.xml
- 2. Make sure youre web.xml has the <distributable/> element
- 3. **All your session attributes must implement java.io.Serializable** 따라서 MemberVO(loginStatus), RSAUtil(rsaPrivateKey) 에 implement를 진행하였다.

# Cluster Basics To run session replication in your Tomcat 8 container, the following steps should be completed: All your session attributes must implement java.io.Serializable Uncomment the Cluster element in server.xml If you have defined custom cluster valves, make sure you have the ReplicationValve defined as well under the Cluster element in server.xml If your Tomcat instances are running on the same machine, make sure the Receiver.port attribute is unique for each instance, in most cases Tomcat is smart enough to resolve this on it's own by autodetecting available ports in the range 4000-4100 Make sure your web.xml has the <distributable/> element If you are using mod jk, make sure that jvmRoute attribute is set at your Engine <Engine name="Catalina" jvmRoute="node01" > and that the jvmRoute attribute value matches your worker name in workers.properties Make sure that all nodes have the same time and sync with NTP service!

### → 각 톰캣을 접속 시 session id 일치 여부 확인

Make sure that your loadbalancer is configured for sticky session mode.





브라우저가 8443, 8444 포트로 각각 다른 톰캣에 접속했음에도 불구하고 Session ID 가 일치하는 것을 볼 수 있다.

또한 nginx를 통해 로그인을 진행한 후 8443, 8444 포트에 따로 접속하는 경우 로그인 상태가 그대로 유지되는 것을 확인할 수 있다.